UNIVERSITÀ DI PARMA
Dipartimento di Ingegneria e Architettura

Firewalls

Luca Veltri

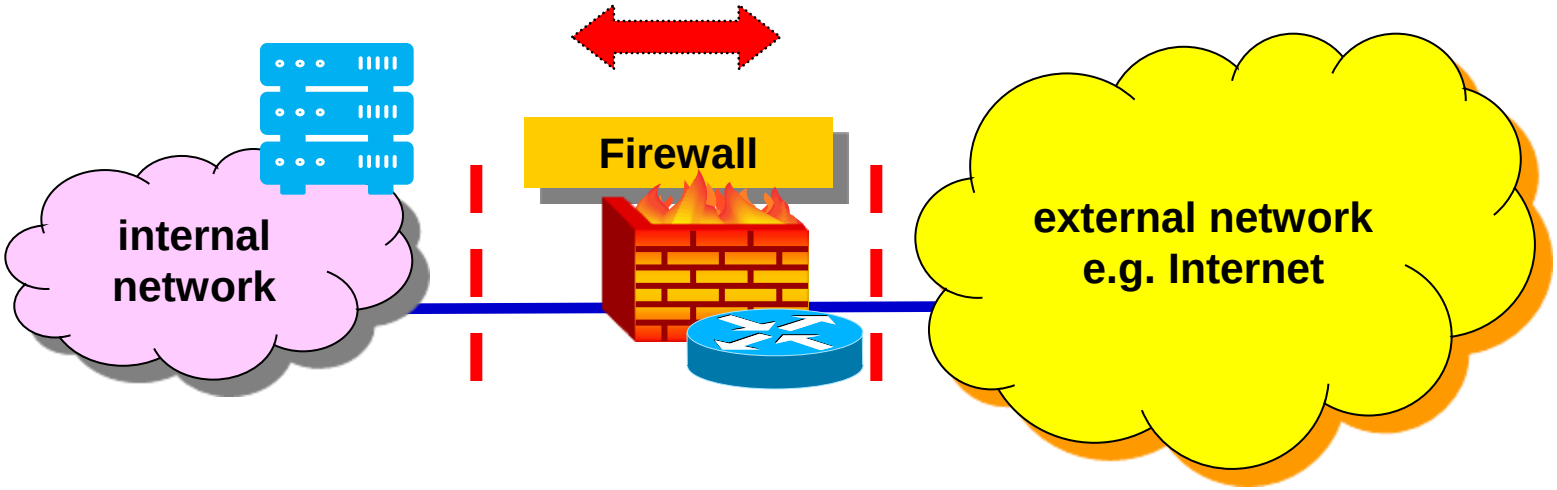
(mail.to: luca.veltri@unipr.it)

Course of Cybersecurity, 2022/2023

http://netsec.unipr.it/veltri

Firewall

- A firewall is security system that controls the incoming and outgoing network traffic
  - **by analyzing the data packets and determining whether they should be allowed through or not, based on a rule set**
  - **builds a bridge between a system (an internal network or computer) and an external network**
  - **can be implemented in HW or SW**



Examples of firewalls

- Many routers that pass data between networks contain firewall components
  - **packet filtering (or screening) router**
- Also intermediate systems working at network or application level may acts as firewall
  - **NAT**
  - **Application Level Gateways (e.g. proxies)**
- Many personal computer OSs include software-based firewalls to protect against threats from the attached network and/or from the public Internet
  - **personal firewalls**
- Example: Linux iptables
    - may work as both personal firewall or router firewall

Firewall Classification

- There are different types of firewalls depending on where the communication is taking place, where the communication is intercepted and the state that is being traced
  - **Network layer or packet filters**
    - Level 3-4 switches
    - Screening routers
  - **Network address translation nodes**
    - NATs and NAPTs
  - **Application-layer firewalls**
    - Host-based application firewalls
    - Network-based application firewalls (proxies)
      - Single-interface bastion hosts
      - Dual-homed systems

When a firewall does not guarantee security

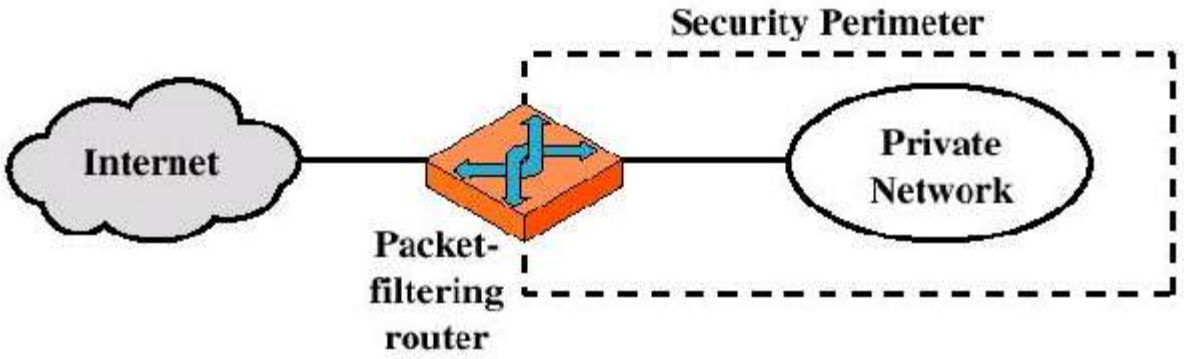
- In case of errors or firewall misconfigurations:
  - **the firewall does not correctly implement the defined security policies**
- The system where the firewall is installed is vulnerable
  - **due to bugs or misconfiguration of the OS or other applications**
- Presence of other paths between the external and internal networks, by-passing the firewall, e.g.
  - **wireless (WiFi) Access Points connected to the internal network, not protected by the firewall**
  - **internal computers/smartphones with 3/4G connections**
- Attacks that start from internal system
  - **malicious users**
  - **caused by compromised mobile computers (laptops, tablets, smarthphones)**
  - **caused by software starting from removable disks / memory sticks**

Packet Filters

Packet filters

- Also referred to as network layer firewalls
- They operate at a relatively low level of the protocol stack, not allowing single packets to pass through the firewall unless they match the established rule set
  - **can filter traffic based on many packet attributes like**
    - source/destination IP addresses,
    - transport protocol,
    - source/destination ports,
    - input/output interfaces,
    - many other packet header fields and attributes
  - **filter rules may be defined by a firewall administrator, or default rules may apply**

Packet Filtering Router



- IP router with filtering capabilities
  - **also referred to as Screening Router**
- Applies a set of rules to each incoming IP packet and then forwards or discards the packet
  - **filter packets going in both directions**
  - **it is typically set up as a list of rules based on matches to fields in the IP and/or transport headers**
  - **two default policies (discard or forward)**

Filtering rules: Example

- Filtering rules are listed in proper lists or tables, sometimes referred to as *Access control lists*, or *Chains*
- Example:
  - **enabling traffic towards and from a mail server (SMTP) and web server (HTTP)**

IP source	*IP dest*	*Proto*	*Sorce port*	*Dest port*	*Action*
*	160.78.1.1	tcp	> 1023	25	permit
*	160.78 .1.2	tcp	> 1023	80	permit
160.78 .1.0/24	*	*	*	*	permit
*	*	*	*	*	deny

Packet Filtering Operation

incoming packet

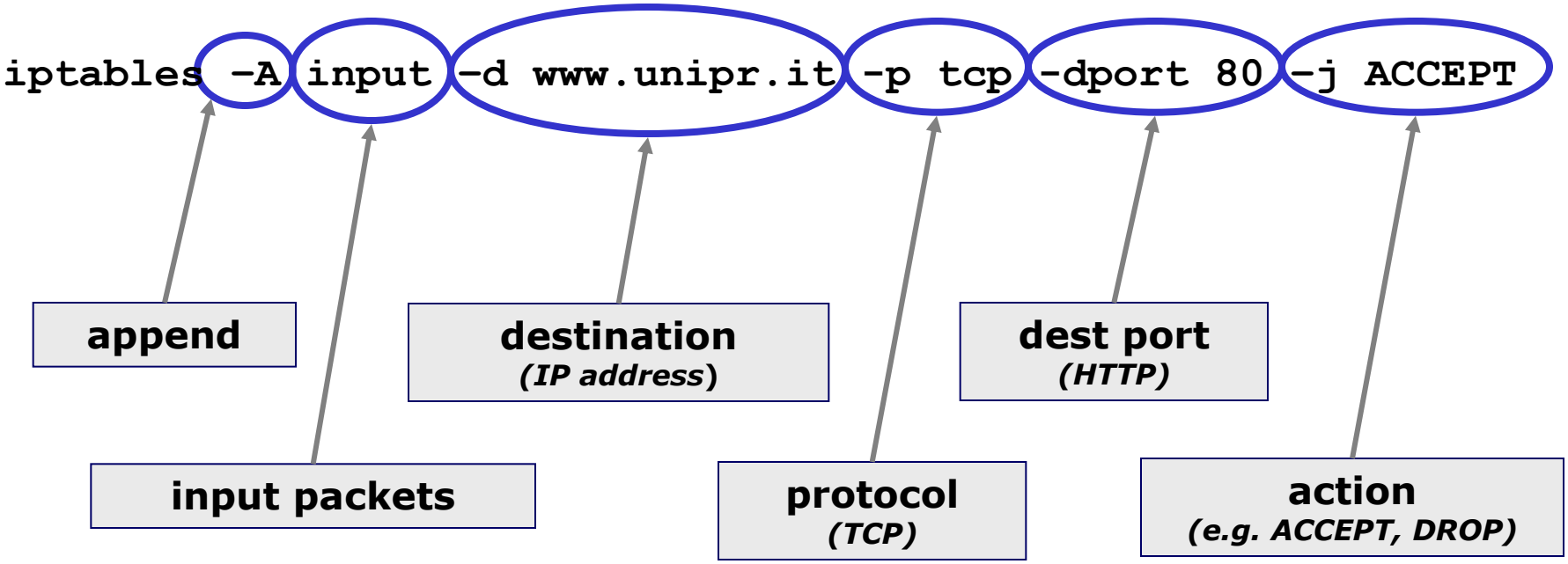
inspection of header and data

enforcing of filtering rules

access control lists

Actions: routing, drop, log, jump to an other list, others

Example of insertion of filtering rules



Stateless vs Stateful packet filters

- Generally fall into two sub-categories: stateless and stateful
  - **Stateless packet filter**
    - decision is per-packet based
      - no state related on previously processed packet
    - require less memory, and can be fast
    - cannot make complex decisions based on what stage communications between hosts have reached
  - **Stateful packet filter**
    - maintain context about active sessions, and use that "state information" to process packets
      - any existing communication is characterized by several properties (source and dest addresses, UDP/TCP ports, connection lifetime, etc.)
      - a firewall's state table is maintained and it contains state (connection) information relate to accepted packets
      - a packet matches an existing connection based on comparison with the state table
      - if a packet does not match an existing connection, it will be evaluated according to the ruleset for new connections

Packet Filters – Advantages/Disadvantages

- Advantages:
  - **Simplicity**
  - **High speed**
  - **Transparency to users**

- Disadvantages:
  - **Difficulty of setting up packet filter rules**
  - **Lack of authentication**

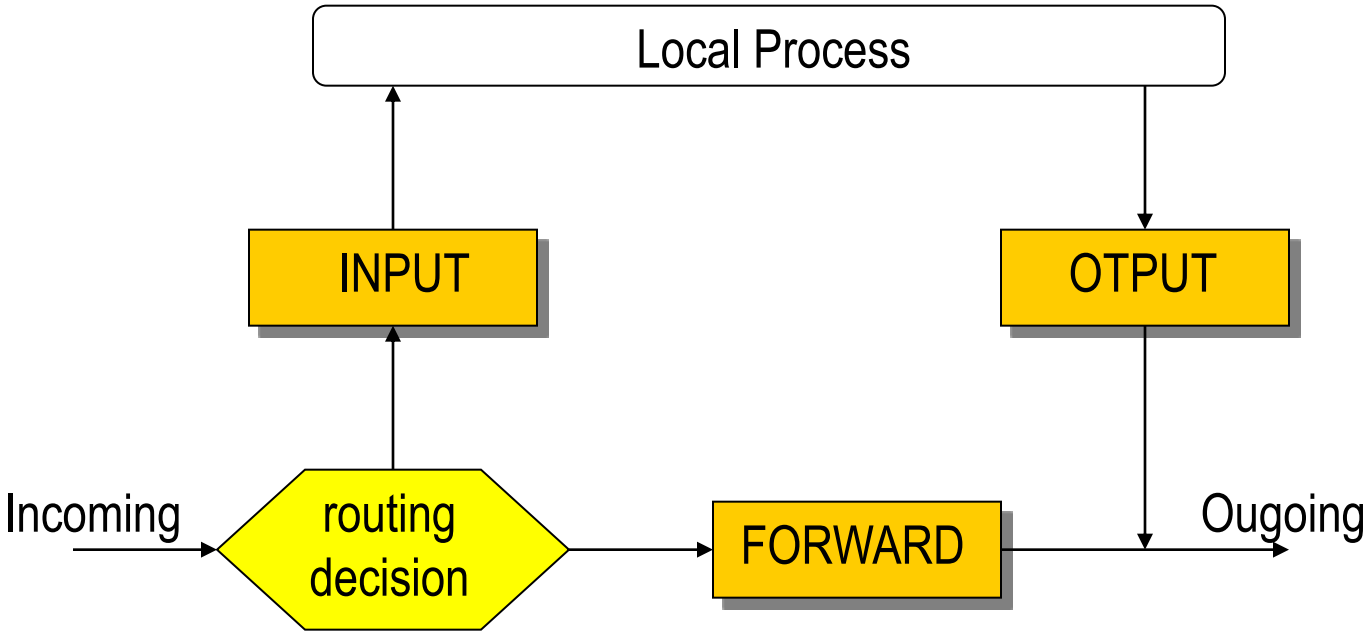
Linux Netfilter (iptables)

Linux packet filter

- Linux kernels have had packet filtering since the 1.1 series
- Packet filtering is implemented by netfilter
- Netfilter is a general framework inside the Linux kernel which other things can plug into
- The tool `iptables` talks to the kernel and tells it what packets to filter
  - **it inserts and deletes rules from the kernel's packet filtering table**
- Lists of filtering rules are called "chains"
  - **that are the Linux's access control lists**

Netfilter basic chains

- The kernel starts with three built-in lists of rules (chains) in the `filter' table
  - **they are INPUT, OUTPUT and FORWARD**
  - **they can't be deleted**



Netfilter basic chains

- When a packet reaches a chain, that chain is examined to decide the fate of the packet
  - **If the chain says to DROP the packet, it is killed there, but**
  - **if the chain says to ACCEPT the packet, it continues traversing the diagram**
- A chain is a checklist of *rules*
  - **each rule says `if the packet header looks like this, then here's what to do with the packet'**
  - **if the rule doesn't match the packet, then the next rule in the chain is consulted**
  - **finally, if there are no more rules to consult, then the kernel looks at the chain policy to decide what to do**
  - **in a security- conscious system, this policy usually tells the kernel to DROP the packet**

`iptables` operations

- Operations to manage whole chains:
  - **Create a new chain (-N)**
  - **Delete an empty chain (-X)**
  - **Change the policy for a built-in chain (-P)**
  - **List the rules in a chain (-L)**
  - **Flush the rules out of a chain (-F)**
  - **Zero the packet and byte counters on all rules in a chain (-Z)**

- Operations to manipulate rules inside a chain:
  - **Append a new rule to a chain (-A)**
  - **Insert a new rule at some position in a chain (-I)**
  - **Replace a rule at some position in a chain (-R)**
  - **Delete a rule at some position in a chain, or the first that matches (-D)**

Managing an entire chain

- Creating a New Chain
  - **using the `-N' (or `--new-chain') command**
  - **e.g. `iptables -N test`**
- Deleting a Chain
  - **using the `-X' (or `--delete-chain') command**
  - **e.g. `iptables -X test`**
- Flushing a Chain
  - **using the `-F' (or `--flush') command**
  - **e.g. `iptables -F FORWARD`**
- Listing a Chain
  - **using the `-L' (or `--list') command**
    - `-n' (numeric) option prevents iptables from to lookup the IP addr
    - `-v' options shows you all the details of the rules

Managing an entire chain (cont.)

- Setting Policy
  - **the policy of the chain determines the default fate of the packet if no rule matches the packet**
  - **only built-in chains (INPUT, OUTPUT and FORWARD) have policies**
  - **The policy can be either ACCEPT or DROP, for example:**
  - **using the `-P' command**
  - **e.g. `# iptables -P FORWARD DROP`**

Managing rules

- Each rule specifies a set of conditions the packet must meet (matching condition),
  and what to do if it meets them (‘target’ or action)
- For example
  - **to drop all ICMP packets coming from the IP address 127.0.0.1**
    - the conditions are that the protocol must be ICMP and that the source address must be 127.0.0.1
    - the target is `DROP’
  - **to add the rule:**

```
iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP
```

matching condition

target

  - **to test:**

```
PING 127.0.0.1
```

  - **to delete the rule:**

```
iptables -D INPUT 1 ,or
iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP
```

Filtering specifications

- Specifying an Interface
  - **the `-i' (or `--in-interface') and `-o' (or `--out-interface') options specify the name of an interface to match**
  - **node that:**
    - INPUT chain don't have an output interface
    - OUTPUT chain don't have an input interface
    - Only FORWARD chain have both an input and output interface
    - an interface name ending with a `+' (wildcard) will match all interfaces which begin with that string
- Specifying Source and Destination IP Addresses
  - **source (`-s', `--source' or `--src') and destination (`-d', `--destination' or `--dst') IP addresses can be specified in four ways**
    - using the full name, such as `localhost' or `www.linuxhq.com'
    - specifying the IP address, such as `127.0.0.1'
    - specifying a group of IP addresses, such as `199.95.207.0/24'
    - or such as `199.95.207.0/255.255.255.0'

Filtering specifications (cont.)

- Specifying Protocol
  - **protocol can be specified with the `-p' (or `--protocol') flag**
  - **protocol can be a number or a name (‘tcp', `udp' or `icmp’)**
- Specifying Inversion
  - **many flags can have their arguments preceded by `!' (NOT) to invert (negate) the given matching condition**
    - e.g. `-s ! localhost' matches any packet not coming from localhost

Matching extensions

- TCP, UDP and ICMP protocols automatically offer specific matching tests
  - **it is possible to specify the new match test on the command line after the `-p' option**
- E.g.

  --source-port (--sport) and --destination-port (--dport)
  - followed by either a single TCP/UDP port, or a range of ports
  - ranges are two port names separated by a `:'
- Other extension can be loaded explicitly
  - **using the `-m' option followed by the match test**
  - **e.g.** `-m mac --mac-source 45:e4:23:6b:82:a0`

The state match

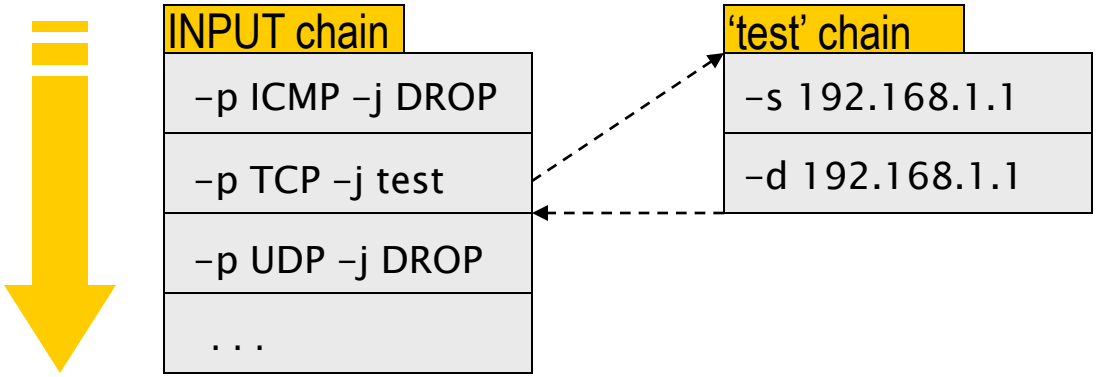
- The '-m state' extension interprets the connection-tracking analysis
  - **connection states are:**
    - NEW
      - a packet which creates a new connection
    - ESTABLISHED
      - a packet which belongs to an existing connection
    - RELATED
      - a packet which is related to, but not part of, an existing connection (e.g. an ICMP error, or an ftp data connection)
    - INVALID
      - a packet which could not be identified for some reason
- Example of '-m state' match extension:
  - `iptables -A FORWARD -i ppp0 -m state --state NEW -j DROP`

Target specifications

- Rule's target is what to do to the packets which match the rule
- There are two very simple built-in targets: DROP and ACCEPT
- Other targets are:
  - **LOG**
    - this module provides kernel logging of matching packets
  - **REJECT**
    - has the same effect as `DROP', except that the sender is sent an ICMP `port unreachable' error message
  - **RETURN**
    - has the same effect of falling off the end of a chain
  - **QUEUE**
    - is a special target, which queues the packet for userspace processing
  - **User-defined chains**

User-defined chains

- It is possible to create new chains, in addition to the three built-in ones (INPUT, FORWARD and OUTPUT)
- When a packet matches a rule whose target is a user-defined chain
  - **the packet begins traversing the rules in that user-defined chain**
  - **if that chain doesn't decide the fate of the packet, then traversal resumes on the next rule in the current chain**



Application-Layer Firewalls

Application-layer firewalls

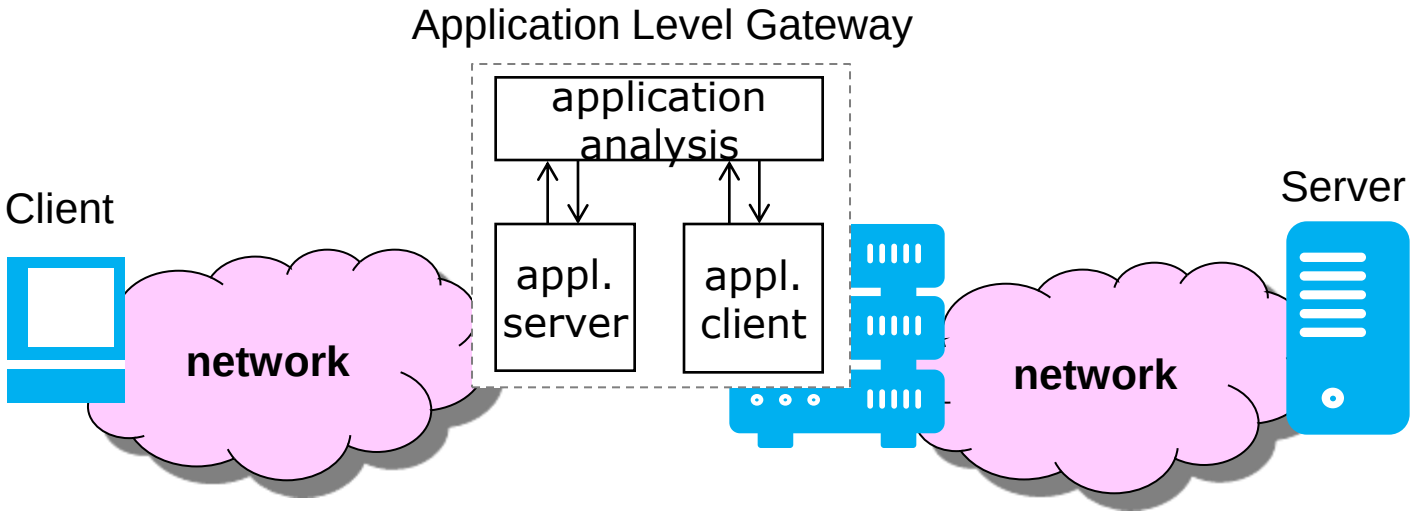
- An application firewall is a form of firewall which controls input, output, and/or access from, to, or by an application or service
  - **can control all network traffic data up to the application layer**
    - it may inspect the contents of traffic, blocking specified content
      - such as certain websites, malicious programs, or attempts to exploit known logical flaws in client software
      - can restrict or prevent the spread of computer malwares
  - **unlike a stateful packet filter firewall which is (without additional software) unable to control network traffic regarding a specific application**
- There are two primary categories of application firewalls
  - **host-based application firewalls**
  - **network-based application firewalls**

Host-based application firewalls

- A host-based application firewall provides protection to the applications running on the same host
- It can monitor any application input, output, and/or system calls made from, to, or by the application
  - **this is done by examining information passed through system calls instead of or in addition to a network stack**
  - **it may block the input, output, or system calls (including socket calls) which do not meet the configured policy of the firewal**
  - **they are able to apply filtering rules (allow/block) on a per process basis instead of filtering connections on a per port basis**
  - **generally, prompts are used to define rules for processes that have not yet received a connection**

Network-based application firewalls

- A network-based application layer firewall operates at the application layer of an (application) intermediate node
  - **also known as proxy-based firewall or application-level gateway**
  - **it acts as a proxy/gateway for specific applications**
    - specific to a particular kind of network traffic
    - e.g. HTTP and FTP proxy, SMTP server, SIP proxy, etc.



Network-based application firewalls (cont.)

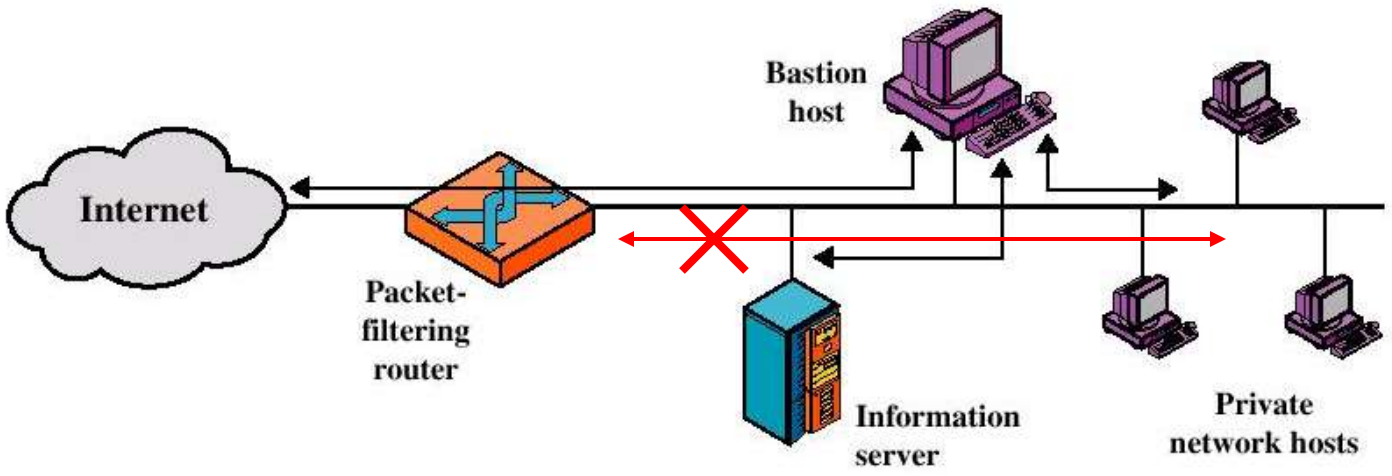
- May run either as a stand-alone piece of network (dedicated) hardware, or as software on a general-purpose machine
  - **often, it is a host using various forms of proxy servers to proxy traffic before passing it on to the client or server**
- May run as single-homed host or on a dual-homed host

Network-based application firewalls (cont.)

- Advantages:
  - **full control at application level**
    - content filtering
    - strong user authentication
    - higher level of security respect to a packet filter
  - **easy to log and audit all incoming traffic**
  - **by default, internal addresses are hidden**
  - **caching**
- Disadvantages:
  - **worse performances**
    - additional processing overhead on each connection
  - **often requires explicit client configurations**
  - **not suitable for new services/applications**
    - does support ONLY services for whjch a corresponding proxy is available (FTP, Telnet, HTTP, SMTP, …)

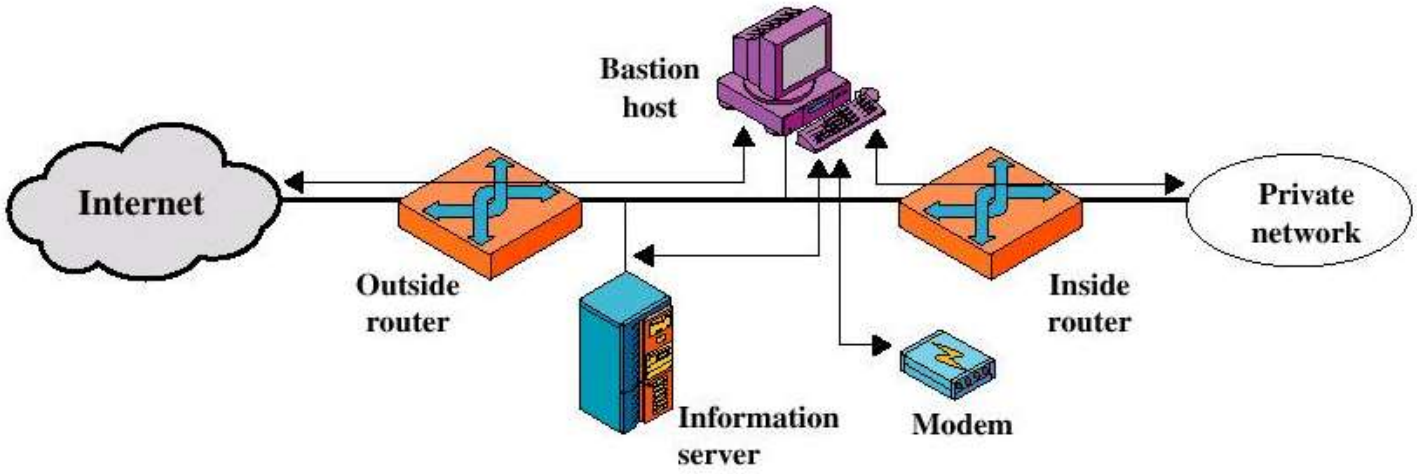
Hybrid configurations

Screened host firewall (single-homed bastion host)



- Screened host firewall, single-homed bastion configuration
- The firewall consists by two systems:
  - **A packet-filtering router**
  - **A bastion host**

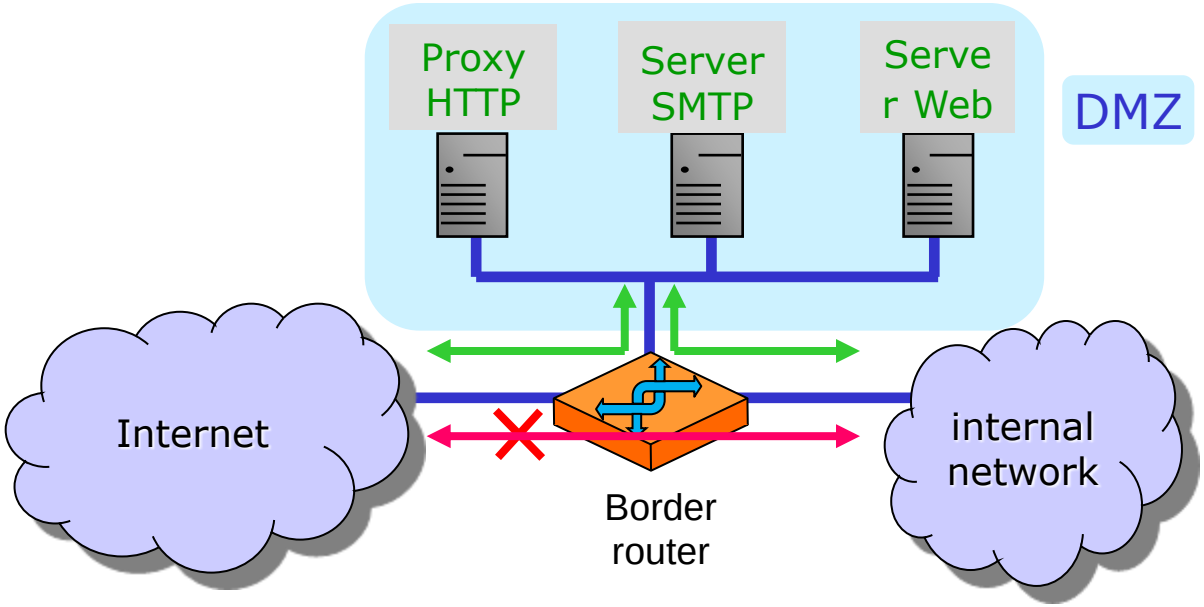
Screened-subnet firewall



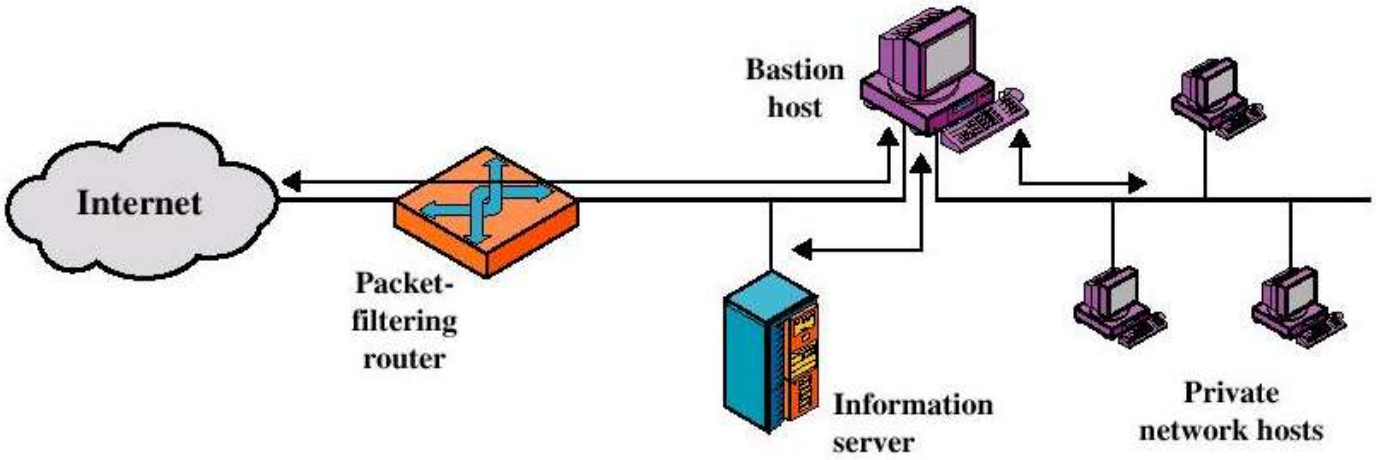
- Screened subnet firewall configuration
  - **Most secure configuration**
  - **Two packet-filtering routers are used**
  - **Creation of an isolated sub-network (DMZ)**

Screened-subnet firewall (cont.)

- Merging of interior/exterior router



Screened host firewall (dual-homed bastion host)



- Screened host firewall, dual-homed bastion configuration
  - **Traffic between the Internet and other hosts on the private network has to flow through the bastion host**
    - regardless the router configuration
  - **If the packet-filtering router is compromised, the network is still not completely compromised**